# NBEMS — a Digital Emcomm Tool

*Dave Kleber, KB3FXI, O'Hara Twp, PA Emergency Management Agency,* **kb3fxi@arrl. net** *and Harry Bloomberg, W3YJ, Assistant SEC, Western Pennsylvania Section,* **w3yj@arrl.net**

COURTESY HARRY BLOOMBERG, W3YJ



**The Narrow Band Emergency Messaging System (NBEMS) in operation on 80 meters.**

## Hurricane Ivan

The remnants of Hurricane Ivan had nearly finished dumping 7 inches of rain on Pittsburgh when Dave Kleber, KB3FXI, received a message. He could expect a couple of busloads of evacuees from flooding in Sharpsburg to begin arriving soon at his position at Parkview Fire Hall. Dave and his crew rapidly began assembling bunk beds and preparing the shelter. Soon, 80 evacuees had arrived.

Not too long afterwards Dave received another message. What were the names, phone numbers and addresses of the people? Landline phones were a mess, the cell phone network was overloaded and the public service radio frequencies were crowded with emergency calls. Amateur Radio was about the only reliable means of communications left.

"Don't they know how long it will take to read all that over the air?" thought Dave.

That was the moment when Dave realized that the traditional Amateur Radio emergency communications model of a ham with a radio at a shelter was no longer good enough.

## Why Digital Emcomm?

The needs of those we serve during disasters and emergencies have changed. We now need to be able to send lists of evacuees in a format compatible with a spreadsheet, inventories of required medical supplies, phone numbers of officials, weather information, directions to an emergency operations center, bulletins of critical situation updates. In other words, we now need to be able to send data not suited to message forms.

The problem is that data is not suited to being relayed by voice. Imagine how long it would take to read a list of evacuees or how hard it would be to spell out phonetically a long list of pharmaceuticals.

So we set out to find a good way to send data. Unfortunately, many of the methods that hams use are either impractical or too expensive for western Pennsylvania, which is very hilly with deep valleys that are difficult for VHF and UHF radio waves. The area does not have much of a digital Amateur Radio infrastructure, so any solution would have to work well with weak signals in valleys and not require an extensive build-out.

## The Solution: MT63 and NBEMS

After several years of experimentation, we hit upon a mode that would work well on VHF and UHF, MT63.

MT63, developed by Pawel Jalocha, SP9VRC, is a very robust mode that transmits data on 64 tones simultaneously in bandwidths of 500 Hz, 1000 Hz or 2000 Hz. It can also be configured to transmit so much redundant data that a 1-2 second gap in a transmission would not lead to a loss of data at the receiving end. MT63 is very forgiving of audio levels, so careful tweaking of volume is not necessary during an emergency. Finally, MT63 works very well in a weak signal environment, so a 2000 Hz wide MT63 signal (abbreviated MT63-2000) could be received deep in our rugged terrain. MT63 also works very well through FM repeaters. Although western Pennsylvania is lacking in digital infrastructure, it has an abundance of repeaters, as there are dozens in the Pittsburgh area.

There was one other property of MT63 that makes it very useful as an emcomm mode. We learned that one can just hold a radio's microphone up to a computer's speakers and be able to transmit MT63. Likewise, at the other end, another ham can hold his radio's speaker up to a computer's microphone and the data will be received by the computer's sound card. In other words, although a radio-to-computer interface like a RIGblaster or SignaLink is great, you really don't need one when you use this method we call "acoustical coupling." This means that you don't have to waste valuable time fumbling around with an octopus of cables or be sidelined by forgetting a cable or an interface.

We now started looking for software that could do MT63. We found one called Narrow Band Emergency Messaging System (NBEMS). NBEMS (**www.w1hkj.com**) consists of two parts: *fldigi,* which acts as a sound card modem and generates audio signals in many different modes in addition to MT63 like MFSK, PSK31 and Olivia; and *flarq,* which allows one to send binary files and place an Automatic Repeat Request (ARQ) layer of handshaking on top of NBEMS. With NBEMS, one can use a repeater channel for either voice or data as needed. There's no need for a channel to be dedicated to just one mode.

There were other features of NBEMS that appealed to us. It ran not only on *Windows XP* and *Vista*, but also *Linux* and Mac. It was easy to install and support. It would transmit PSK31 and RTTY signals so it could be used for recreational nonemcomm hamming. And it was *free*. NBEMS is released under the GNU Public License (GPL), which means that it is unencumbered by patent or restrictive licenses. It also means that you receive the original source code for the program and you're free to

modify the code as you see fit, so you're protected against a vendor going out of business or changing the terms of software use.

NBEMS also allowed us to easily send bulletins to large numbers of stations at one time. With connected networks like packet radio, a bulletin station can send messages to only one station at a time, or to put it another way, there is a one-to-one relationship between the station originating the bulletin and receiving stations. But any station monitoring a channel containing NBEMS data would receive the bulletin. There is a one-to-many relationship between sender and receiver. One can therefore use NBEMS to transmit situation updates, weather information, road closures and other critical information to many stations at the same time. Stations receiving the data can be unattended. All digital NBEMS data transmitted will be captured by *fldigi* and can be examined at the convenience of the operator in the field.

The combination of NBEMS, MT63 and acoustical coupling gave us a powerful, flexible, relatively simple means of sending and receiving data. We now had a digital emcomm package that required only a ham, any VHF/UHF FM radio and nearly any computer. No need for external modems, TNCs nor dedicated digital mode radios; no modified radios; interfacing between radio and computer was no longer a headache; the software was "free"; we did not need to procure, set up, configure and maintain a dedicated digital infrastructure, and we did not need to worry about an incident occurring outside the range of any specialized digital network.

## Was the Message Received?

There was still one obstacle for us to hurdle before NBEMS could be adopted. How do you know that a particular message had been received 100%? MT63 is very robust, but for some critical emergency traffic, one needs to know for certain that a message was received. With many shorter text messages, a ham can just "eyeball" the message and see that something's amiss. But how can one determine that data exported from a spreadsheet, was received intact? The *flarq* program can do this, but only at a cost of significant overhead.

We proposed a solution to the problem to the NBEMS developers. How about embedding a checksum in the data sent by *fldigi* that could be used by the receiving station to determine if the message had been received intact? A checksum is the result of a calculation involving all of the data in the message. If the receiving station computed the same checksum in the message as the sending station, the message had been received 100%. This is not as elegant a solution as *flarq*, but it was more efficient and simpler to implement.

After some discussion, the developers went to work and in a very short period of time came up with a solution, a program called *Wrap*. *Wrap* would envelop a message with special strings to indicate the start and end of "wrapped" data. Within this wrapped message would be stored a checksum and the name of a file.

This wrapped message would then be sent using *fldigi*. At the receiving end, *fldigi* would look for the start and end of the wrapped data. Once a wrapped message had been identified, *fldigi* would extract it and store it in a folder. The operator would then run *Wrap* against this file to compute a checksum on the received data and to compare it to the checksum that the sending station had embedded in the message. If the two checksums were equal, the receiving operator would see a message indicating success and the original message would be extracted.

## Training

We then started an effort to recruit and train operators in using NBEMS. Dave organized an informal group named wpaNBEMS (**www.wpanbems.org**). This group holds training nets on repeaters using Olivia and MT63.

Harry volunteered as an Assistant Section Emergency Coordinator and wrote a western Pennsylvania digital emcomm standards document and helped train ARES® groups.

One recent training net illustrated the portability of the NBEMS software and the fact that it does not require very powerful or expensive hardware. We were able to successfully send and receive MT63 data between machines running Windows Vista, Windows XP, Macintosh OS X and Ubuntu Linux. Several of our net members use Dell Mini 9s for NBEMS, running both Linux and Windows XP, and find that they work very well as a cheap digital emcomm workstation.

## Applying Lessons Learned

Dave and Harry participated in the 2008 Simulated Emergency Test. Harry was assigned to a parking lot of UPMC St Margaret Hospital in Pittsburgh. Dave was located at a simulated EOC at Skyview Radio Society's clubhouse approximately 10 miles away.

Harry was asked to use NBEMS to send a text file exported from a spreadsheet. This spreadsheet contained the names of 25 patients sent to the hospital, and for each patient, a nine digit ID number, a phone number and a postal address.

Just as Harry was about to begin transmitting the data, the repeater that had been chosen for the data transmission failed. No big deal. Harry and Dave met on a backup repeater and the data file was successfully transmitted in just under 2 minutes. The harsh lesson that Ivan taught us had been learned.